Pakistan Suicide Bombing Attacks (1995 - 2016)

01. Uploading the Dataset:-

In [ ]:
```
import pandas as pd
bombari_df = pd.read_csv("PakistanSuicideAttacks Ver 11 (30-November-2017).csv",encoding='cp1252')
bombari_df
```

Out[ ]:

	S#	Date	Islamic Date	Blast Day Type	Holiday Type	Time	City	Latitude	Longitude	Province	...	Targeted Sect if any	Killed Min	Killed Max	Injured Min	Injured Max	No. of Suicide Blasts	Explosive Weight (max)	Hospital Names
0	1	Sunday-November 19-1995	25 Jumaada al-THaany 1416 A.H	Holiday	Weekend	NaN	Islamabad	33.718000	73.0718	Capital	...	NaN	14.0	15.0	NaN	60	2.0	NaN	NaN
1	2	Monday-November 6-2000	10 SHa`baan 1421 A.H	Working Day	NaN	NaN	Karachi	24.991800	66.9911	Sindh	...	NaN	NaN	3.0	NaN	3	1.0	NaN	NaN
2	3	Wednesday-May 8-2002	25 safar 1423 A.H	Working Day	NaN	7:45 AM	Karachi	24.991800	66.9911	Sindh	...	Christian	13.0	15.0	20.0	40	1.0	2.5 Kg	1.Jinnah Postgraduate Medical Center 2. Civil ...
3	4	Friday-June 14-2002	3 Raby` al-THaany 1423 A.H	Working Day	NaN	11:10:00 AM	Karachi	24.991800	66.9911	Sindh	...	Christian	NaN	12.0	NaN	51	1.0	NaN	NaN
4	5	Friday-July 4-2003	4 Jumaada al-awal 1424 A.H	Working Day	NaN	NaN	Quetta	30.209500	67.0182	Baluchistan	...	Shiite	44.0	47.0	NaN	65	1.0	NaN	1.CMH Quetta \n2.Civil Hospital 3. Boland Medi...
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
491	492	Thursday-October-05-2017	13 MuHarram 1439 A.H	Working Day	NaN	NaN	Quetta	28.571051	67.496895	Balochistan	...	NaN	21.0	22.0	25.0	NaN	1.0	NaN	Larkana Hospital, Gandawah District Headquarte...
492	493	Thursday-October-19-2017	27 MuHarram 1439 A.H	Working Day	NaN	8:00 AM	Quetta	NaN	NaN	Balochistan	...	NaN	7.0	7.0	22.0	22	1.0	NaN	NaN
493	494	Thursday-November-9-2017	19 Safar 1439 A.H	Working Day	NaN	7:00 AM	Quetta	30.221057	67.002524	Balochistan	...	NaN	2.0	2.0	8.0	8	1.0	15KG	Combined Military Hospital (CMH)-(Quetta), Civ...
494	495	Friday-November-24-2017	5 Rabi Al-Awwal 1439 A.H	Working Day	NaN	7:00 AM	Peshawar	33.970623	71.43862	KPK	...	NaN	1.0	3.0	6.0	8	1.0	20KG	Hayatabad Medical Complex(Peshawar)
495	496	Saturday-November-25-2017	6 Rabi Al-Awwal 1439 A.H	Working Day	NaN	7:00 AM	Quetta	30.139626	66.981476	Balochistan	...	NaN	4.0	4.0	16.0	22	1.0	10KG	Civil Hospital(Quetta)

496 rows × 26 columns

02. Data Cleaning & Preparation:-

checking columns of the dataset:-

In [ ]:
```
bombari_df.columns
```

Out[ ]:
```
Index(['S#', 'Date', 'Islamic Date', 'Blast Day Type', 'Holiday Type', 'Time',
       'City', 'Latitude', 'Longitude', 'Province', 'Location',
       'Location Category', 'Location Sensitivity', 'Open/Closed Space',
       'Influencing Event/Event', 'Target Type', 'Targeted Sect if any',
       'Killed Min', 'Killed Max', 'Injured Min', 'Injured Max',
       'No. of Suicide Blasts', 'Explosive Weight (max)', 'Hospital Names',
       'Temperature(C)', 'Temperature(F)'],
      dtype='object')
```

checking basic information of the dataset:-

In [ ]:
```
bombari_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 496 entries, 0 to 495
Data columns (total 26 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   S#                       496 non-null    int64
 1   Date                     496 non-null    object
 2   Islamic Date             342 non-null    object
 3   Blast Day Type           486 non-null    object
 4   Holiday Type             72 non-null     object
 5   Time                     285 non-null    object
 6   City                     496 non-null    object
 7   Latitude                 493 non-null    float64
 8   Longitude                493 non-null    object
 9   Province                 496 non-null    object
 10  Location                 493 non-null    object
 11  Location Category        461 non-null    object
 12  Location Sensitivity     460 non-null    object
 13  Open/Closed Space        461 non-null    object
 14  Influencing Event/Event  191 non-null    object
 15  Target Type              470 non-null    object
 16  Targeted Sect if any     97 non-null     object
 17  Killed Min               350 non-null    float64
 18  Killed Max               480 non-null    float64
 19  Injured Min              365 non-null    float64
 20  Injured Max              464 non-null    object
 21  No. of Suicide Blasts    414 non-null    float64
 22  Explosive Weight (max)   172 non-null    object
 23  Hospital Names           297 non-null    object
 24  Temperature(C)           491 non-null    float64
 25  Temperature(F)           489 non-null    float64
dtypes: float64(7), int64(1), object(18)
memory usage: 100.9+ KB
```

correcting the data types of the columns:-

In [ ]:
```
bombari_df['Killed Min'] = bombari_df['Killed Min'].astype('Int64')
bombari_df['Killed Max'] = bombari_df['Killed Max'].astype('Int64')
```

```
bombari_df['Injured Min'] = bombari_df['Injured Min'].astype('Int64')
bombari_df['Injured Max'] = pd.to_numeric(bombari_df["Injured Max"], errors= 'coerce')
bombari_df['Injured Max'] = bombari_df['Injured Max'].astype('Int64')
bombari_df['No. of Suicide Blasts'] = bombari_df['No. of Suicide Blasts'].astype('Int64')
```

getting only 'years' from the 'date column' Because they can be more useful in visualizations:-

In [ ]:
```
bombari_df['Year'] = bombari_df.Date.str[-4:]
```

Extracting 'Islamic Months' from the 'Islamic Date' column Because they can be more useful in visualizations:-

In [ ]:
```
bombari_df['Islamic Date'].isnull().sum()
```

Out[ ]:
```
154
```

In [ ]:
```
bombari_df['Islamic Date'] = bombari_df['Islamic Date'].fillna('')
```

In [ ]:
```
bombari_df["Islamic Date"] = bombari_df["Islamic Date"].replace("","date unknown_data month islamic year A.H")
bombari_df["Islamic Months"] = bombari_df["Islamic Date"].apply(lambda x : x.split()[1])


bombari_df['Islamic Months'].unique()
```

Out[ ]:
```
array(['Jumaada', 'SHa`baan', 'safar', 'Raby`', 'Thw', 'MuHarram',
       'unknown_data', 'Rajab', 'SHawwal', 'RamaDHaan', 'Shaban', 'Zeqad',
       'Zilhaj', 'Jamadi-ul-Awal', 'Jumada', 'Shawaal', 'Safar', 'Rabi'],
      dtype=object)
```

replacing with the correct islamic month names:-

In [ ]:
```
bombari_df["Islamic Months"] = bombari_df["Islamic Months"].replace( {"Jumaada" :  "Jumaada al-Thaany", "Jumada":  "Jumaada al-Thaany",
                              "MuHarram": "Muharram", "safar": "Safar", "Rabi": "Rabi-ul-Awwal", "Raby": "Rabi-ul-Thaany",
                              "Jamadi-ul-Awal": "Jamadi-ul-Awwal", "SHabaan": "Shaban", "SHa`baan": 'Shaban', "RamaDHaan" : "Ramazan",
                              "SHawwal": "Swawwal", "Thw": "Zilhaj"})


bombari_df['Islamic Months'].unique()
```

Out[ ]:
```
array(['Jumaada al-Thaany', 'Shaban', 'Safar', 'Raby`', 'Zilhaj',
       'Muharram', 'unknown_data', 'Rajab', 'Swawwal', 'Ramazan', 'Zeqad',
       'Jamadi-ul-Awwal', 'Shawaal', 'Rabi-ul-Awwal'], dtype=object)
```

checking 'Blast Day Type' column:-

In [ ]:
```
bombari_df['Blast Day Type'].unique()
```

Out[ ]:
```
array(['Holiday', 'Working Day', nan, 'Weekend'], dtype=object)
```

checking 'Holiday Type' column:-

In [ ]:
```
bombari_df['Holiday Type'].unique()
```

Out[ ]:
```
array(['Weekend', nan, 'Christmas/birthday of Quaid-e-Azam', 'Ashura',
       'Eid Milad un-Nabi', 'Iqbal Day', 'Eid-ul-azha', 'Labour Day',
       'Eid-ul-Fitar', 'Pakistan Day', 'Defence Day',
       'Christmas/ birthday of Quaid-e-Azam', 'General Elections',
       'Eid Holidays', 'Ashura Holiday', 'Eid ul Azha Holiday'],
      dtype=object)
```

making data more accurate:-

In [ ]:
```
bombari_df['Holiday Type'] = bombari_df['Holiday Type'].replace({'Christmas/ birthday of Quaid-e-Azam': 'Christmas/birthday of Quaid-e-Azam',
                'Ashura': 'Ashura Holiday', 'Eid ul Azha Holiday': 'Eid-ul-azha', 'Eid-ul-azha': 'Eid Holidays', 'Eid-ul-Fitar':'Eid Holidays',
                'Eid-ul-azha': 'Eid Holidays'})

bombari_df['Holiday Type'].unique()
```

Out[ ]:
```
array(['Weekend', nan, 'Christmas/birthday of Quaid-e-Azam',
       'Ashura Holiday', 'Eid Milad un-Nabi', 'Iqbal Day', 'Eid Holidays',
       'Labour Day', 'Pakistan Day', 'Defence Day', 'General Elections',
       'Eid-ul-azha'], dtype=object)
```

dealing with the 'City' Column:-

In [ ]:
```
bombari_df['City'].unique()
```

Out[ ]:
```
array(['Islamabad', 'Karachi', 'Karachi ', 'Quetta', 'Rawalpindi',
       'North waziristan', 'Kohat', 'Attock ', 'Sialkot', 'Lahore ',
       'Swat', 'Hangu', 'Bannu', 'North Waziristan', 'Lasbela',
       'Malakand ', 'Peshawar', 'Peshawar ', 'D.I Khan ', 'Lakki Marwat',
       'Tank', 'Islamabad ', 'Quetta ', 'Gujrat', 'Charsadda', 'Tank ',
       'Swat ', 'D.I Khan', 'Kohat ', 'Kuram Agency ', 'Shangla ',
       'Bajaur Agency', 'South Waziristan', 'Haripur', 'bannu',
       'karachi ', 'Sargodha', 'Nowshehra', 'Charsadda ',
       'Mohmand agency', 'Dara Adam Khel', 'Khyber Agency', 'Mardan',
       'Bhakkar ', 'Orakzai Agency', 'Mohmand Agency', 'Buner',
       'D.G Khan ', 'Pishin', 'Chakwal ', 'Lahore', 'Upper Dir',
       'Muzaffarabad', 'swat', 'Totalai', 'Multan', 'D.G Khan',
       'Lower Dir', 'Lakki marwat', 'Sudhanoti', 'Khyber Agency ',
       'peshawar', 'Poonch', 'Mansehra', 'Karak', 'Mohmand Agency ',
       'South waziristan', 'lakki marwat', 'Swabi ', 'Dara Adam khel',
       'karachi', 'Shikarpur', 'Sukkur', 'Chaman', 'D. I Khan',
       'Khanewal', 'Fateh Jang', 'Taftan', 'Tirah Valley', 'Wagah',
       'Zhob', 'KURRAM AGENCY', 'ATTOCK', 'MULTAN', 'Taunsa', 'Jacobabad',
       'Shabqadar-Charsadda', 'Khuzdar', 'Ghallanai, Mohmand Agency ',
       'Hayatabad', 'Mosal Kor, Mohmand Agency', 'Sehwan town',
       'Tangi, Charsadda District'], dtype=object)
```

making data consistent & accurate:-

In [ ]:
```
bombari_df['City'] = bombari_df['City'].str.strip().str.lower()
cities = bombari_df['City'].unique()
# cities
```

In [ ]:
```
# say no to warnings...
import warnings
warnings.filterwarnings('ignore')
```

In [ ]:
```
import numpy as np
from fuzzywuzzy import fuzz

# Create an empty list to store the similar strings
similar_strings = []

# Loop through each city in the array
for city in cities:
```

```
    # Loop through the rest of the cities
    for other_city in cities:
        # Calculate the Levenshtein distance between the two strings
        distance = fuzz.ratio(city, other_city)
        # If the distance is greater than or equal to 80 (i.e. they are very similar), and they are not the same string
        if distance >= 80 and city != other_city:
            # Add the pair of similar strings to the list
            if sorted((city, other_city)) not in similar_strings:
                similar_strings.append(sorted((city, other_city)))

similar_strings
```

Out[ ]:
```
[['north waziristan', 'south waziristan'],
 ['d.g khan', 'd.i khan'],
 ['d. i khan', 'd.i khan'],
 ['kuram agency', 'kurram agency'],
 ['d. i khan', 'd.g khan']]
```

replacing with the correct City names:-

In [ ]:
```
bombari_df.City = bombari_df.City.replace({'d. i khan': 'd.i khan', 'kuram agency': 'kurram agency'})
```

Results:-

In [ ]:
```
bombari_df.City.unique()
```

Out[ ]:
```
array(['islamabad', 'karachi', 'quetta', 'rawalpindi', 'north waziristan',
       'kohat', 'attock', 'sialkot', 'lahore', 'swat', 'hangu', 'bannu',
       'lasbela', 'malakand', 'peshawar', 'd.i khan', 'lakki marwat',
       'tank', 'gujrat', 'charsadda', 'kurram agency', 'shangla',
       'bajaur agency', 'south waziristan', 'haripur', 'sargodha',
       'nowshehra', 'mohmand agency', 'dara adam khel', 'khyber agency',
       'mardan', 'bhakkar', 'orakzai agency', 'buner', 'd.g khan',
       'pishin', 'chakwal', 'upper dir', 'muzaffarabad', 'totalai',
       'multan', 'lower dir', 'sudhanoti', 'poonch', 'mansehra', 'karak',
       'swabi', 'shikarpur', 'sukkur', 'chaman', 'khanewal', 'fateh jang',
       'taftan', 'tirah valley', 'wagah', 'zhob', 'taunsa', 'jacobabad',
       'shabqadar-charsadda', 'khuzdar', 'ghallanai, mohmand agency',
       'hayatabad', 'mosal kor, mohmand agency', 'sehwan town',
       'tangi, charsadda district'], dtype=object)
```

checking 'Province' column:-

In [ ]:
```
bombari_df.Province.unique()
```

Out[ ]:
```
array(['Capital', 'Sindh', 'Baluchistan', 'Punjab', 'FATA', 'KPK', 'AJK',
       'Fata', 'Balochistan'], dtype=object)
```

replacing with the correct names:-

In [ ]:
```
bombari_df.Province = bombari_df.Province.replace({'Baluchistan': 'Balochistan', 'Fata': 'FATA'})

bombari_df.Province.unique()
```

Out[ ]:
```
array(['Capital', 'Sindh', 'Balochistan', 'Punjab', 'FATA', 'KPK', 'AJK'],
      dtype=object)
```

checking 'Location' column:-

In [ ]:
```
bombari_df.Location.sample(15)
```

Out[ ]:
```
436                                        Imambargah
390                                Bolan Medical Complex
290                                      WFP office Khar
405                 Torkhan customs office in Landikotal
48                                Training Center in Hangu
298                                        Batakhel area
368    Two suicide bombers blew themselves up at a cr...
269                     FC chowk near Dean's trade center
146                near a check-post in Officers' Colony
160    Mosque in Jamrood Tehsil on peshawar-turkham h...
16                                     Shrine of Bari Imam
284    Mohmad Agency-Ghalani area-Political \nAgent o...
305                        Near DCO office-Nowshehra road
181              Quetta-Khi RCD highway-town of Torkham
262                         Bilalabad area near Khar town
Name: Location, dtype: object
```

checking 'Location Category' column:-

In [ ]:
```
bombari_df['Location Category'].unique()
```

Out[ ]:
```
array(['Foreign', 'Office Building', 'Hotel', 'Religious', 'Mobile',
       'Military', 'Residence', 'Park/Ground', 'Market', 'Police',
       'Airport', 'Government', 'Transport', 'Hospital', 'Educational',
       'Civilian', 'Bank', 'Foreigner', 'Government/Office Building',
       'Government Official', 'foreign', nan, 'Residential Building',
       'Commercial/residence', ' ', 'Highway'], dtype=object)
```

making data accurate:-

In [ ]:
```
bombari_df['Location Category'] = bombari_df['Location Category'].replace({'Foreigner': 'Foreign', 'foreign': 'Foreign',
                                'Government/Office Building': 'Office Building', 'Residential Building': 'Residence', ' ': np.nan})

bombari_df['Location Category'].unique()
```

Out[ ]:
```
array(['Foreign', 'Office Building', 'Hotel', 'Religious', 'Mobile',
       'Military', 'Residence', 'Park/Ground', 'Market', 'Police',
       'Airport', 'Government', 'Transport', 'Hospital', 'Educational',
       'Civilian', 'Bank', 'Government Official', nan,
       'Commercial/residence', 'Highway'], dtype=object)
```

overview of all columns:-

In [ ]:
```
bombari_df.columns
```

Out[ ]:
```
Index(['S#', 'Date', 'Islamic Date', 'Blast Day Type', 'Holiday Type', 'Time',
       'City', 'Latitude', 'Longitude', 'Province', 'Location',
       'Location Category', 'Location Sensitivity', 'Open/Closed Space',
       'Influencing Event/Event', 'Target Type', 'Targeted Sect if any',
       'Killed Min', 'Killed Max', 'Injured Min', 'Injured Max',
       'No. of Suicide Blasts', 'Explosive Weight (max)', 'Hospital Names',
       'Temperature(C)', 'Temperature(F)', 'Year', 'Islamic Months'],
      dtype='object')
```

checking 'Location Sensitivity' column:-

In [ ]:
```
bombari_df['Location Sensitivity'].unique()
```

Out[ ]:
```
array(['High', 'Low', 'Medium', nan, 'low'], dtype=object)
```

making data accurate:-

In [ ]:
```
bombari_df['Location Sensitivity'] = bombari_df['Location Sensitivity'].replace({'low': 'Low'})

bombari_df['Location Sensitivity'].unique()
```

Out[ ]:
```
array(['High', 'Low', 'Medium', nan], dtype=object)
```

checking 'Open/Closed Space' column:-

In [ ]:
```
bombari_df['Open/Closed Space'].unique()
```

Out[ ]:
```
array(['Closed', 'Open', 'open', 'Open ', nan, 'closed', 'Open/Closed'],
      dtype=object)
```

making data accurate:-

In [ ]:
```
bombari_df['Open/Closed Space'] = bombari_df['Open/Closed Space'].str.strip()
bombari_df['Open/Closed Space'] = bombari_df['Open/Closed Space'].replace({'open': 'Open', 'closed': 'Closed'})

bombari_df['Open/Closed Space'].unique()
```

Out[ ]:
```
array(['Closed', 'Open', nan, 'Open/Closed'], dtype=object)
```

checking 'Influencing Event/Event' column:-

In [ ]:
```
bombari_df['Influencing Event/Event'].sample(15)
```

Out[ ]:
```
301                                            NaN
473    Attack on Dargah Shah Noorani in Khuzdar district
166                                            NaN
253                    Leader of anti-militants lashkar
227                                            NaN
249                                            NaN
356                                            NaN
459                              Attack on Polio Centre
404    It is said that Bannu TTP Militant commander S...
484    Terrorist ambushed a police party while it was...
478    inauguration of the Out Patient Department (OP...
254                                            NaN
313                                            NaN
465                                            NaN
12                                       Friday prayer
Name: Influencing Event/Event, dtype: object
```

checking 'Target Type' column:-

In [ ]:
```
bombari_df['Target Type'].unique()
```

Out[ ]:
```
array(['Foreigner', 'Media', 'Religious', 'Military',
       'Government Official', 'civilian', 'Civilian', 'Police',
       'Government official', 'police', 'Children/Women',
       'Anti-Militants', 'foreigner', 'Unknown', 'religious', nan,
       'Rangers', 'Police & Rangers', 'Civilian & Police', 'Army',
       'Frontier Corps ', 'advocates (lawyers)', 'Civilian Judges',
       'Shia sect', 'Judges & lawyers'], dtype=object)
```

making data more accurate:-

In [ ]:
```
bombari_df['Target Type'] = bombari_df['Target Type'].str.lower().str.strip()
bombari_df['Target Type'] = bombari_df['Target Type'].replace({'civilian judges': 'judges & lawyers', 'advocates (lawyers)': 'judges & lawyers',
                                'police': 'police & rangers', 'rangers': 'police & rangers'})

bombari_df['Target Type'].unique()
```

Out[ ]:
```
array(['foreigner', 'media', 'religious', 'military',
       'government official', 'civilian', 'police & rangers',
       'children/women', 'anti-militants', 'unknown', nan,
       'civilian & police', 'army', 'frontier corps', 'judges & lawyers',
       'shia sect'], dtype=object)
```

checking 'Targeted Sect if any' column:-

In [ ]:
```
bombari_df['Targeted Sect if any'].unique()
```

Out[ ]:
```
array([nan, 'Christian', 'Shiite', 'shiite', 'Shiite/sunni', 'Sunni',
       'Jews', 'Ahmedi'], dtype=object)
```

making data more accurate:-

In [ ]:
```
bombari_df['Targeted Sect if any'] = bombari_df['Targeted Sect if any'].replace({'shiite': 'Shiite'})

bombari_df['Targeted Sect if any'].unique()
```

Out[ ]:
```
array([nan, 'Christian', 'Shiite', 'Shiite/sunni', 'Sunni', 'Jews',
       'Ahmedi'], dtype=object)
```

checking 'Explosive Weight (max)' column:-

In [ ]:
```
bombari_df['Explosive Weight (max)'].unique()
```

Out[ ]:
```
array([nan, '2.5 Kg', '30kg in each car', '2kg', '10-15 kg', '5 kg',
       '10 kg ', '5 to 6 Kg', '10 to 15 kg', 'more than 5Kg', '5 to 6 kg',
       '15 kg', '3 kg explosive and 3kg ball bearings', '6-7kg', '4-5kg',
       '50kg', '30kg ', '100Kg', '150Kg', '1000 Kg', ' 8 Kg', '60Kg',
       '8 ft deep crater', '40 kg', '7 or 8Kg', ' 10 kg ', '8Kg', '100kg',
       '100', '85', '125 kg', '5kg', '160', '500 Kg', '24Kg', '10Kg',
       '20 Kg', '12Kg', '12 Kg', '150 kg ', '180Kg', '100 Kg', '70 kg ',
       '16 Kg', '15Kg', '10 Kg', '7Kg ', '400Kg', '200Kg', '50-60Kg',
       '250Kg', '1000 KG', '250 kg', '600 kg', '40Kg', '5Kg', ' 12 kg',
       '600Kg', '12 kg', '14 kg', '8 kg', '300 kg', '20 kg ', '10 kg',
       '20Kg', ' 6 kg', '1200Kg', '12 kg ', '1000 Kg ', '400 KG',
       '1000 kg', '7Kg', '9Kg', '16Kg', '250kg', '8kg', '180kg',
       '16-20kg', '300kg', '450kg', '7kg', '60kg', '500 kg', '5-6kg',
       '3kg', '6kg', '6Kg', '100-110kg', '30kg', '7kg-8kg', '100kg-150kg',
       '4kg-5kg', '12kg', '600kg-800kg', '4kg-6kg', '1000kg', '7-8kg',
       '6-7Kg', '20-25kg', '25-30kg', '14kg', '10kg', '45KG', '15KG',
       '13KG', '8KG', '8-10KG', '6KG', '8-10 KG', ' 20 KG', '15 KG',
       '75KG', '25KG', '20KG', '10KG'], dtype=object)
```

making data consistent & accurate:-

In [ ]:
```
bombari_df['Explosive Weight (max)'] = bombari_df['Explosive Weight (max)'].str.strip()
bombari_df['Explosive Weight (max)'] = bombari_df['Explosive Weight (max)'].str.replace('KG', 'kg')
bombari_df['Explosive Weight (max)'] = bombari_df['Explosive Weight (max)'].str.replace('Kg', 'kg')
bombari_df['Explosive Weight (max)'] = bombari_df['Explosive Weight (max)'].str.replace(' ', '')

bombari_df['Explosive Weight (max)'] = bombari_df['Explosive Weight (max)'].replace({'30kgineachcar': '30kg', '3kgexplosiveand3kgballbearings': '6kg',
                          'morethan5kg': '6kg', '7kg-8kg': '7-8kg', '100kg-150kg': '100-150kg', '4kg-5kg': '4-5kg', '600kg-800kg': '600-800kg',
                          '4kg-6kg': '4-6kg', '5to6kg': '6kg', '10to15kg': '12kg', '7or8kg': '7kg'})

bombari_df['Explosive Weight (max)'].replace('8ftdeepcrater', np.nan, inplace=True)

# bombari_df['Explosive Weight (max)'].unique()
```

In [ ]:
```
result = [s for s in bombari_df['Explosive Weight (max)'].unique() if '-' in str(s)]
result
```

Out[ ]:
```
['10-15kg',
 '6-7kg',
 '4-5kg',
 '50-60kg',
 '16-20kg',
 '5-6kg',
 '100-110kg',
 '7-8kg',
 '100-150kg',
 '600-800kg',
 '4-6kg',
 '20-25kg',
 '25-30kg',
 '8-10kg']
```

In [ ]:
```
bombari_df['Explosive Weight (max)'] = bombari_df['Explosive Weight (max)'].replace({

'10-15kg': '12kg',
'6-7kg': '6kg',
'4-5kg': '4kg',
'50-60kg': '55kg',
'16-20kg': '18kg',
'5-6kg': '5kg',
'100-110kg': '105kg',
'7-8kg': '7kg',
'100-150kg': '125kg',
'600-800kg': '700kg',
'4-6kg': '5kg',
'20-25kg': '22kg',
'25-30kg': '27kg',
'8-10kg': '9kg'
})
```

Results:-

In [ ]:
```
bombari_df['Explosive Weight (max)'].unique()
```

Out[ ]:
```
array([nan, '2.5kg', '30kg', '2kg', '12kg', '5kg', '10kg', '6kg', '15kg',
       '4kg', '50kg', '100kg', '150kg', '1000kg', '8kg', '60kg', '40kg',
       '7kg', '100', '85', '125kg', '160', '500kg', '24kg', '20kg',
       '180kg', '70kg', '16kg', '400kg', '200kg', '55kg', '250kg',
       '600kg', '14kg', '300kg', '1200kg', '9kg', '18kg', '450kg', '3kg',
       '105kg', '700kg', '22kg', '27kg', '45kg', '13kg', '75kg', '25kg'],
      dtype=object)
```

checking 'Temperature(C)' colummn:-

In [ ]:
```
bombari_df['Temperature(C)']
```

Out[ ]:
```
0      15.835
1      23.770
2      31.460
3      31.430
4      33.120
        ...
491    27.800
492    26.000
493    25.000
494    22.000
495    29.000
Name: Temperature(C), Length: 496, dtype: float64
```

making data accurate:-

In [ ]:
```
bombari_df = bombari_df.replace([np.inf, -np.inf], np.nan).dropna(subset=['Temperature(C)'])
bombari_df['Temperature(C)'] = bombari_df['Temperature(C)'].round().astype(int)
```

In [ ]:
```
bombari_df['Temperature(C)']
```

Out[ ]:
```
0      16
1      24
2      31
3      31
4      33
       ..
491    28
492    26
493    25
494    22
495    29
Name: Temperature(C), Length: 491, dtype: int32
```

making 'Hospital Names' column data more consistent:-

In [ ]:
```
bombari_df['Hospital Names'].head(10)
```

Out[ ]:
```
0                                            NaN
1                                            NaN
2    1.Jinnah Postgraduate Medical Center 2. Civil ...
3                                            NaN
4    1.CMH Quetta \n2.Civil Hospital 3. Boland Medi...
5                  1.District headquarters \nHospital
6                                            NaN
7                                   Civil hospital
8    Civil hospital-Liaquat National hospital-Jinna...
9                                            NaN
Name: Hospital Names, dtype: object
```

let's look at our dataset now:-

In [ ]:
```
bombari_df
```

Out[ ]:

	S#	Date	Islamic Date	Blast Day Type	Holiday Type	Time	City	Latitude	Longitude	Province	...	Killed Max	Injured Min	Injured Max	No. of Suicide Blasts	Explosive Weight (max)	Hospital Names	Temperature(C)	T
0	1	Sunday-November 19-1995	25 Jumaada al-THaany 1416 A.H	Holiday	Weekend	NaN	islamabad	33.718000	73.0718	Capital	...	15	<NA>	60	2	NaN	NaN	16	
1	2	Monday-November 6-2000	10 SHa`baan 1421 A.H	Working Day	NaN	NaN	karachi	24.991800	66.9911	Sindh	...	3	<NA>	3	1	NaN	NaN	24	
2	3	Wednesday-May 8-2002	25 safar 1423 A.H	Working Day	NaN	7:45 AM	karachi	24.991800	66.9911	Sindh	...	15	20	40	1	2.5kg	1.Jinnah Postgraduate Medical Center 2. Civil ...	31	
3	4	Friday-June 14-2002	3 Raby` al-THaany 1423 A.H	Working Day	NaN	11:10:00 AM	karachi	24.991800	66.9911	Sindh	...	12	<NA>	51	1	NaN	NaN	31	
4	5	Friday-July 4-2003	4 Jumaada al-awal 1424 A.H	Working Day	NaN	NaN	quetta	30.209500	67.0182	Balochistan	...	47	<NA>	65	1	NaN	1.CMH Quetta \n2.Civil Hospital 3. Boland Medi...	33	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
491	492	Thursday-October-05-2017	13 MuHarram 1439 A.H	Working Day	NaN	NaN	quetta	28.571051	67.496895	Balochistan	...	22	25	<NA>	1	NaN	Larkana Hospital, Gandawah District Headquarte...	28	
492	493	Thursday-October-19-2017	27 MuHarram 1439 A.H	Working Day	NaN	8:00 AM	quetta	NaN	NaN	Balochistan	...	7	22	22	1	NaN	NaN	26	
493	494	Thursday-November-9-2017	19 Safar 1439 A.H	Working Day	NaN	7:00 AM	quetta	30.221057	67.002524	Balochistan	...	2	8	8	1	15kg	Combined Military Hospital (CMH)-(Quetta), Civ...	25	
494	495	Friday-November-24-2017	5 Rabi Al-Awwal 1439 A.H	Working Day	NaN	7:00 AM	peshawar	33.970623	71.43862	KPK	...	3	6	8	1	20kg	Hayatabad Medical Complex(Peshawar)	22	
495	496	Saturday-November-25-2017	6 Rabi Al-Awwal 1439 A.H	Working Day	NaN	7:00 AM	quetta	30.139626	66.981476	Balochistan	...	4	16	22	1	10kg	Civil Hospital(Quetta)	29	

491 rows × 28 columns

03. Exploratory Analysis & Visualizations:-

In [ ]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

__columns overview:-__

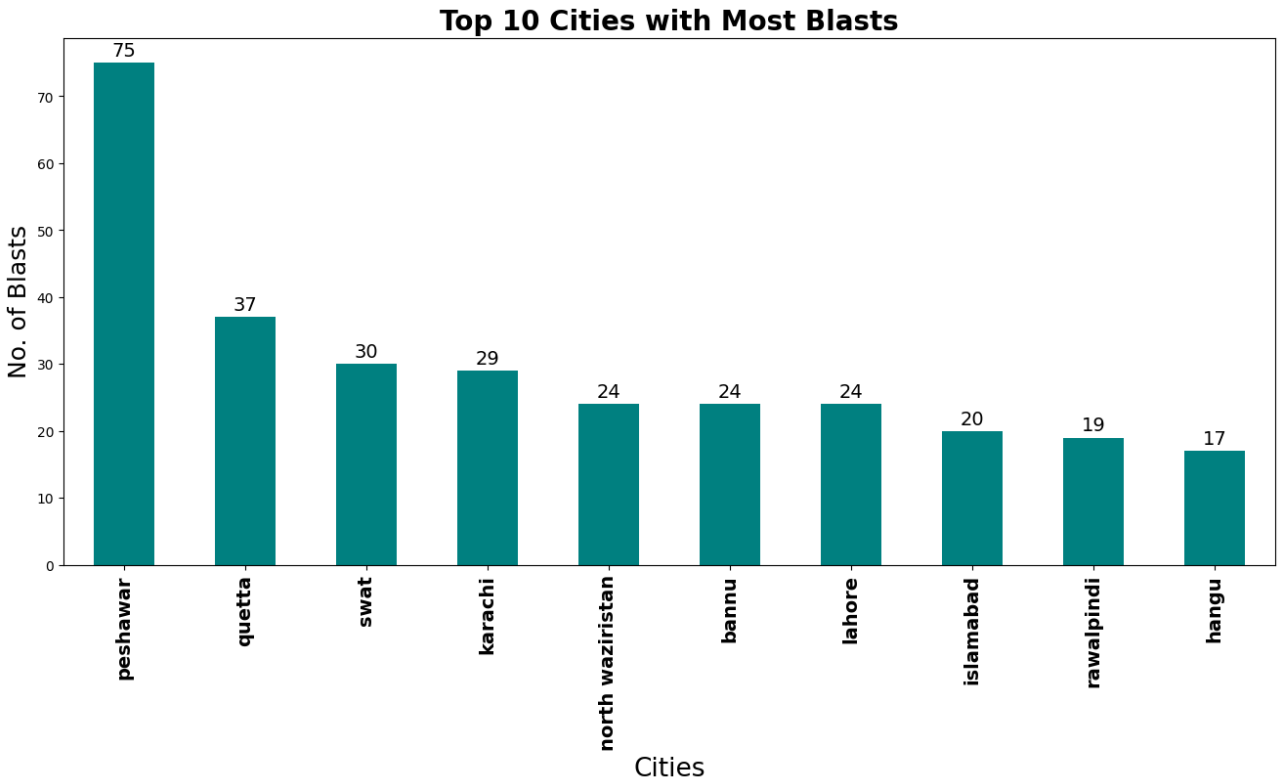
In [ ]:
```
bombari_df.columns
```

Out[ ]:
```
Index(['S#', 'Date', 'Islamic Date', 'Blast Day Type', 'Holiday Type', 'Time',
       'City', 'Latitude', 'Longitude', 'Province', 'Location',
       'Location Category', 'Location Sensitivity', 'Open/Closed Space',
       'Influencing Event/Event', 'Target Type', 'Targeted Sect if any',
       'Killed Min', 'Killed Max', 'Injured Min', 'Injured Max',
       'No. of Suicide Blasts', 'Explosive Weight (max)', 'Hospital Names',
       'Temperature(C)', 'Temperature(F)', 'Year', 'Islamic Months'],
      dtype='object')
```

01. Top 10 Cities with Most Blasts:-

__plotting:-__

In [ ]:
```
plot_1 = bombari_df['City'].value_counts().head(10)

plt.figure(figsize=(16, 7))
plot_1.plot(kind='bar', color='teal')
plt.title('Top 10 Cities with Most Blasts', fontsize=20, fontweight='bold')
plt.xlabel('Cities', fontsize=19)
plt.ylabel('No. of Blasts', fontsize=18)

for i in range(len(plot_1)):
    plt.text(i, plot_1[i]+1, plot_1[i], ha='center', fontsize=14)

plt.xticks(range(len(plot_1)), plot_1.index, fontsize=14, fontweight='bold')

plt.show()
```



02. Sucide Blasts Locations in Pakistan:-

extracting pakistan's latitudes & longitudes for the map:-

In [ ]:
```
import geopandas as gpd
from shapely.geometry import mapping

world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))

poly = world[world.name == "Pakistan"].geometry
poly_mapped = mapping(poly)

poly_coordinates = poly_mapped["features"][0]['geometry']['coordinates'][0]
poly_dict =  [{'lat': coords[1],'lon': coords[0]} for coords in poly_coordinates]
pak_boundary_coord = pd.DataFrame(poly_dict)
```

In [ ]:
```
pak_boundary_coord
```

Out[ ]:

	lat	lon
0	35.494010	77.837451
1	34.653544	76.871722
2	34.504923	75.757061
3	34.748887	74.240203
4	34.317699	73.749948
...	...	...
61	37.020841	74.575893
62	37.133031	75.158028
63	36.666806	75.896897
64	35.898403	76.192848
65	35.494010	77.837451

66 rows × 2 columns

overview:-

In [ ]:
```
# fig = px.Line(pak_boundary_coord, x="lon", y="lat", height=800, width=1300)
# fig.update_traces(line_color="darkgreen")
```

checking city names:-

In [ ]:
```
bombari_df.City.unique()
```

Out[ ]:
```
array(['islamabad', 'karachi', 'quetta', 'rawalpindi', 'north waziristan',
       'kohat', 'attock', 'sialkot', 'lahore', 'swat', 'hangu', 'bannu',
       'lasbela', 'malakand', 'peshawar', 'd.i khan', 'lakki marwat',
       'tank', 'gujrat', 'charsadda', 'kurram agency', 'shangla',
       'bajaur agency', 'south waziristan', 'haripur', 'sargodha',
       'nowshehra', 'mohmand agency', 'dara adam khel', 'khyber agency',
       'mardan', 'bhakkar', 'orakzai agency', 'buner', 'd.g khan',
       'pishin', 'chakwal', 'upper dir', 'muzaffarabad', 'totalai',
       'multan', 'lower dir', 'sudhanoti', 'poonch', 'mansehra', 'karak',
       'swabi', 'shikarpur', 'sukkur', 'chaman', 'khanewal', 'fateh jang',
       'taftan', 'tirah valley', 'wagah', 'zhob', 'taunsa', 'jacobabad',
       'shabqadar-charsadda', 'ghallanai, mohmand agency', 'hayatabad',
       'sehwan town', 'tangi, charsadda district'], dtype=object)
```

feeding correct latitudes and longitudes for the above cities in the dataset:-

In [ ]:
```
city_dict = {'islamabad': (33.6844, 73.0479),
'karachi': (24.8607, 67.0011),
```

```
'quetta': (30.1798, 66.9750),
'rawalpindi': (33.5651, 73.0169),
'north waziristan': (32.9881, 69.9344),
'kohat': (33.5869, 71.4417),
'attock': (33.7717, 72.3592),
'sialkot': (32.4947, 74.5319),
'lahore': (31.5820, 74.3294),
'swat': (34.7566, 72.3558),
'hangu': (33.5292, 71.0728),
'bannu': (32.9869, 70.6050),
'lasbela': (26.8531, 66.3833),
'malakand': (34.5500, 71.9167),
'peshawar': (34.0150, 71.5250),
'd.i khan': (31.8297, 70.9017),
'lakki marwat': (32.6144, 70.9125),
'tank': (32.2219, 70.3797),
'gujrat': (32.5722, 74.0750),
'charsadda': (34.1450, 71.7406),
'kurram agency': (33.8813, 69.8922),
'shangla': (34.7953, 72.4753),
'bajaur agency': (34.8103, 71.8158),
'south waziristan': (32.3617, 69.8553),
'haripur': (33.9990, 72.9343),
'sargodha': (32.0836, 72.6711),
'nowshehra': (34.0158, 71.9819),
'mohmand agency': (34.1658, 71.9244),
'dara adam khel': (33.8439, 71.8864),
'khyber agency': (34.0511, 71.1083),
"mardan": (34.1987, 72.0499),
"bhakkar": (31.6267, 71.0656),
"orakzai agency": (33.1025, 70.1394),
"buner": (34.5278, 72.4786),
"d.g khan": (30.0561, 70.6342),
"pishin": (30.5806, 66.9961),
"chakwal": (32.9308, 72.8583),
"upper dir": (35.2058, 71.8761),
"muzaffarabad": (34.3700, 73.4711),
"totalai": (34.6342, 72.6522),
"multan": (30.1989, 71.4783),
"lower dir": (34.8156, 71.8342),
"sudhanoti": (33.7947, 73.5836),
"poonch": (33.7669, 74.1070),
"mansehra": (34.3334, 73.1968),
"karak": (33.1167, 71.0833),
"swabi": (34.1229, 72.4726),
"shikarpur": (27.9555, 68.6375),
"sukkur": (27.7136, 68.8486),
"chaman": (30.9186, 66.4464),
"khanewal": (30.3012, 71.9329),
"fateh jang": (33.3918, 72.9776),
"taftan": (29.7979, 61.7027),
"tirah valley": (33.7466, 70.1176),
"wagah": (31.6030, 74.4403),
"zhob": (31.3416, 69.4490),
"taunsa": (30.6975, 70.6722),
"jacobabad": (28.2810, 68.4390),
"shabqadar-charsadda": (34.1354, 71.8096),
"khuzdar": (27.7384, 66.6432),
"ghallanai, mohmand agency": (34.2224, 71.5558),
"hayatabad": (33.9902, 71.4623),
"mosal kor, mohmand agency": (34.2639, 71.5397),
"sehwan town": (26.4241, 67.8609),
"tangi, charsadda district": (34.1835, 71.7525)}

for index, row in bombari_df.iterrows():
    # get the name of the city for the current row
    city = row['City']
    # get the latitude and longitude for the current city from the city_dict
    lat, long = city_dict[city]
    # replace the latitude and longitude values in the current row with the lat and long values
    bombari_df.at[index, 'Latitude'] = lat
    bombari_df.at[index, 'Longitude'] = long
```

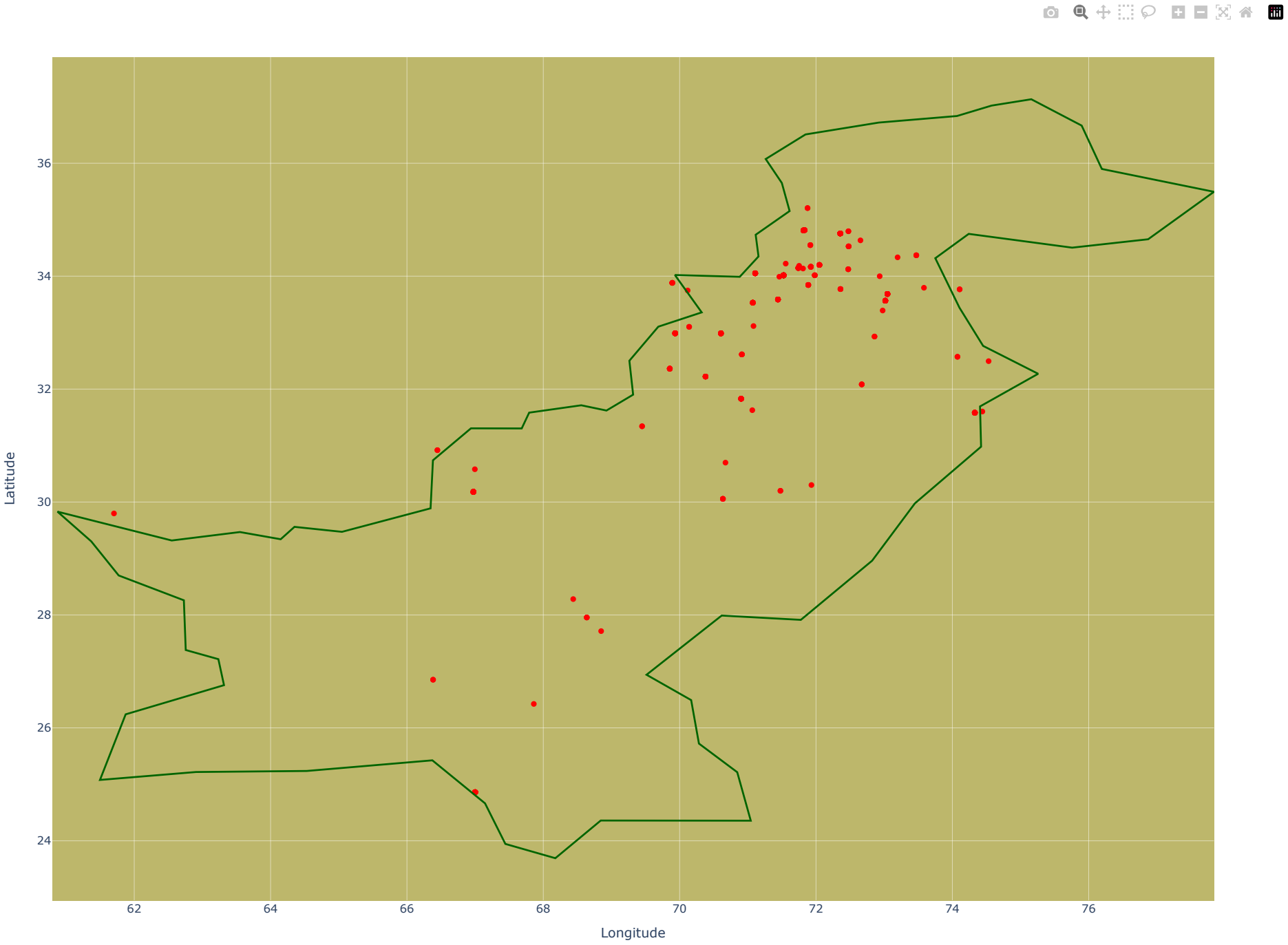
__plotting:-__

In [ ]:
```
import plotly.graph_objects as go
import plotly.offline as pyo

fig = px.scatter(bombari_df, x='Longitude', y='Latitude', hover_name='City', hover_data=['Date', 'Province'])
fig.update_traces(marker=dict(color='red', size=6), hoverlabel=dict(font=dict(family='Arial', size=18, color='white')))
fig.update_layout(mapbox_style="open-street-map")
fig.update_layout(mapbox_zoom=5, mapbox_center = {"lat": 30.3753, "lon": 69.3451})
fig.add_trace(px.line(pak_boundary_coord, x="lon", y="lat").data[0])
fig.update_traces(line_color='darkgreen')
fig.update_layout(height=1040, width=1400, plot_bgcolor='darkkhaki')

pyo.plot(fig, filename='my_plot.html', auto_open=True)

fig.update_traces(hovertemplate="<b>Blast Location: %{hovertext}</b><br>" +
                                 "<b>Date:</b> %{customdata[0]}<br>" +
                                 "<b>Province:</b> %{customdata[1]}<br>" +
                                 "<extra></extra>")

fig.update_layout(
    title={
        'text': 'Sucide Blasts Locations in Pakistan',
        'font': {'size': 30, 'color': 'black'},
        'x': 0.5,
        'xanchor': 'center'
    },
    xaxis_title='Longitude',
    xaxis_title_font={'size': 22, 'color': 'black'},
    yaxis_title_font={'size': 22, 'color': 'black'}
)

display(fig)
```

03. Average No. of People Killed & Injured per Year:-

__dealing:-__

In [ ]:
```
bombari_df['Average Killed'] = bombari_df['Killed Max'] + bombari_df['Killed Min'] / 2
bombari_df['Average Injured'] = bombari_df['Injured Max'] + bombari_df['Injured Min'] / 2

bombari_df['Average Killed'].fillna(0, inplace=True)
bombari_df['Average Injured'].fillna(0, inplace=True)
```


36
34
32
30
28
26
24
Latitude
62
64
66
68
70
72
74
76
Longitude

```
bombari_df['Average Killed'] = bombari_df['Average Killed'].astype('int')
bombari_df['Average Injured'] = bombari_df['Average Injured'].astype('int')
```

grouping:-

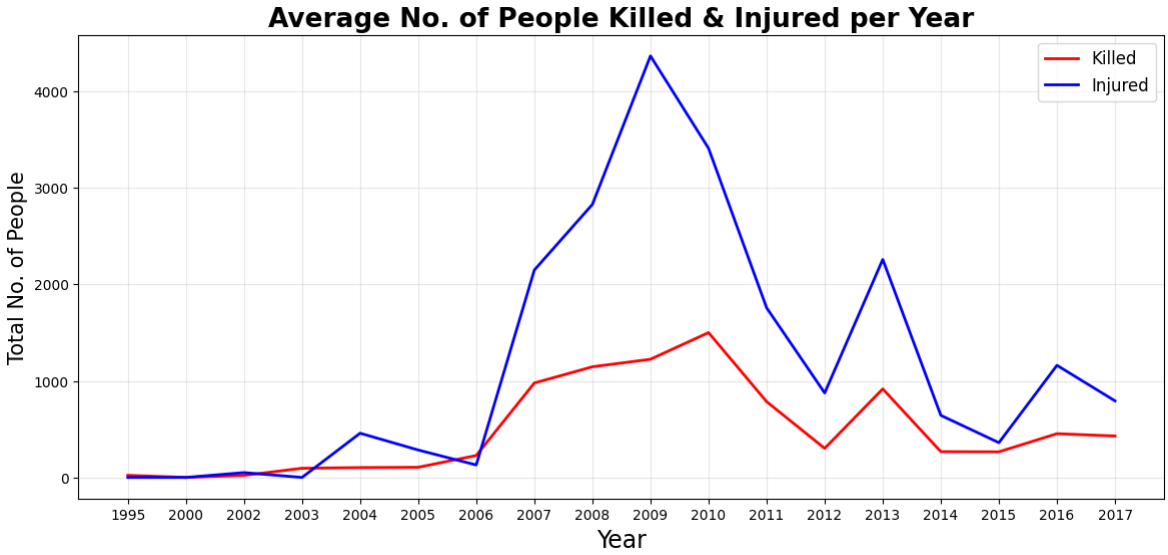
In [ ]:
```
result = bombari_df.groupby("Year")[['Average Killed', 'Average Injured']].sum()
result
```

Out[ ]:

Year	Average Killed	Average Injured
1995	22	0
2000	0	0
2002	21	50
2003	95	0
2004	102	458
2005	105	284
2006	227	129
2007	978	2148
2008	1146	2828
2009	1224	4366
2010	1500	3409
2011	785	1757
2012	301	875
2013	917	2258
2014	267	643
2015	265	360
2016	453	1161
2017	429	793

plotting:-

In [ ]:
```
plt.figure(figsize=(14,6))

plt.plot(result.index, result['Average Killed'], label='Killed', color='red', linewidth=2)
plt.plot(result.index, result['Average Injured'], label='Injured', color='blue', linewidth=2)

plt.xlabel('Year', fontsize=17)
plt.ylabel('Total No. of People', fontsize=15)
plt.title('Average No. of People Killed & Injured per Year', fontsize=19, fontweight = 'bold')
plt.legend(fontsize=12)
plt.grid(alpha=0.3)
plt.show()
```

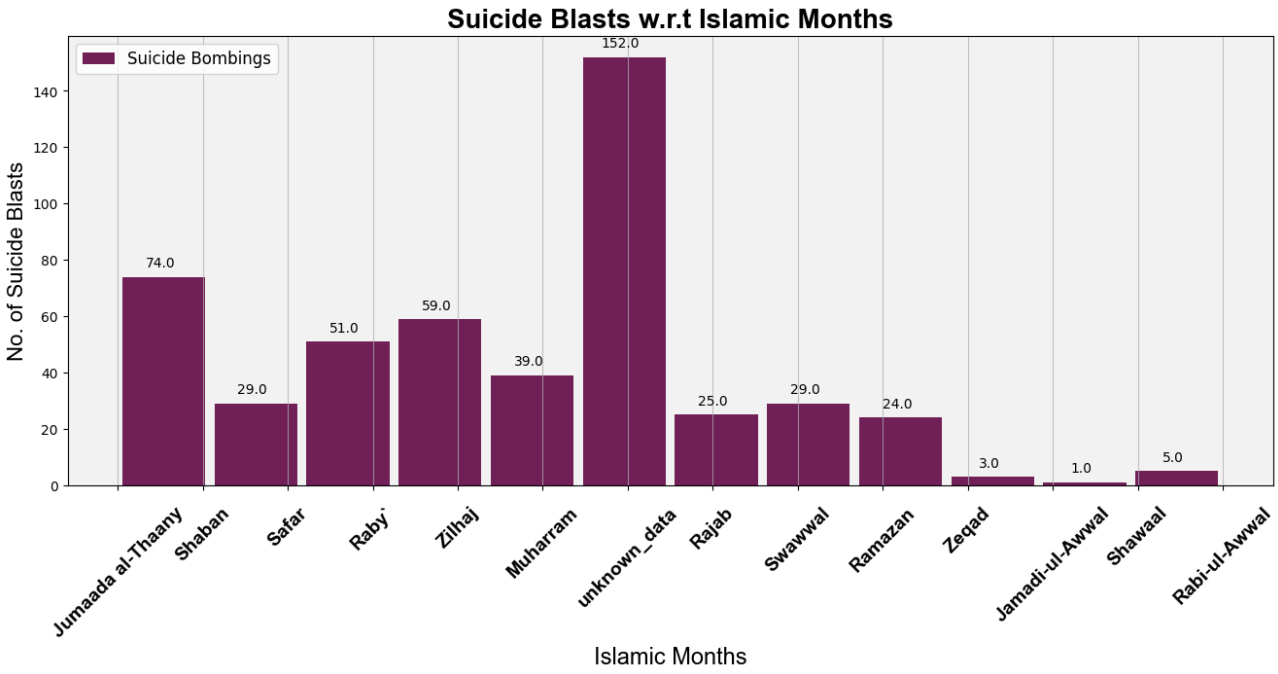


04. Suicide Blasts w.r.t Islamic Months:-

plotting:-

In [ ]:
```
import matplotlib.ticker as ticker

plt.figure(figsize=(16, 6))

colors = sns.color_palette('rocket')
counts, edges, patches = plt.hist(bombari_df['Islamic Months'], bins=12, rwidth=0.9, color=colors[1])

for count, edge, patch in zip(counts, edges, patches):
    if count > 0:
        plt.annotate(str(count), xy=(edge + 0.5, count), xytext=(0, 5), textcoords='offset points', ha='center', va='bottom')

plt.grid(axis='x', alpha=0.75)
plt.gca().set_facecolor('#f2f2f2')
plt.title('Suicide Blasts w.r.t Islamic Months', fontsize=20, fontweight='bold', fontname='Arial')
plt.xlabel('Islamic Months', fontsize=17, fontname='Arial')
plt.xticks(fontsize=14, fontweight='bold', rotation=45, fontname='Arial')
plt.ylabel('No. of Suicide Blasts', fontsize=16, fontname='Arial')
plt.tick_params(axis='x', which='major', pad=10)
plt.gca().yaxis.set_major_formatter(ticker.FormatStrFormatter('%1.0f'))
plt.subplots_adjust(wspace=0.4)
plt.legend(['Suicide Bombings'], loc='upper left', fontsize=12)
```

```
plt.show()
```



05. Percentage of Suicide Blasts By Location Sensitivity:

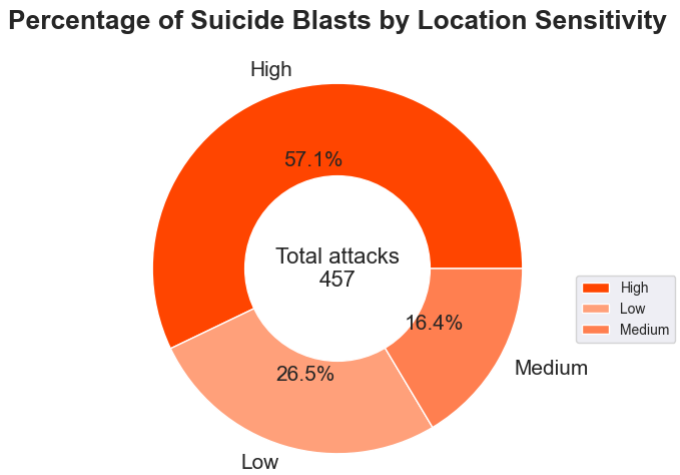
dealing:-

In [ ]:
```
df_loc_sens = bombari_df.groupby(['Location Sensitivity']).size().reset_index(name='Count')

# Calculate the percentage of attacks for each location sensitivity
total_attacks = df_loc_sens['Count'].sum()
df_loc_sens['Percentage'] = df_loc_sens['Count'] / total_attacks * 100
```

plotting:-

In [ ]:
```
plt.figure(figsize= (17,6))

sns.set_style("darkgrid")
colors = ['orangered', 'lightsalmon', 'coral']

fig, ax = plt.subplots(figsize=(8, 6))
ax.pie(df_loc_sens['Percentage'], labels=df_loc_sens['Location Sensitivity'], autopct='%1.1f%%', colors=colors, wedgeprops=dict(width=0.6, edgecolor='white'), textprops={'fontsi

ax.set_title('Percentage of Suicide Blasts by Location Sensitivity', fontsize=19, fontweight = 'bold')
ax.legend(loc='best', bbox_to_anchor=(1, 0.5))

circle = plt.Circle(xy=(0,0), radius=0.5, facecolor='white')
ax.add_artist(circle)

total_attacks_str = f"Total attacks\n{total_attacks}"
ax.text(0, 0, total_attacks_str, ha='center', va='center', fontsize=16)

ax.set_xticks([])
ax.set_yticks([])
ax.set_xlabel('')
ax.set_ylabel('')

plt.show()
```

```
<Figure size 1700x600 with 0 Axes>
```



06. Total Victims w.r.t Target Type:-

dealing:-

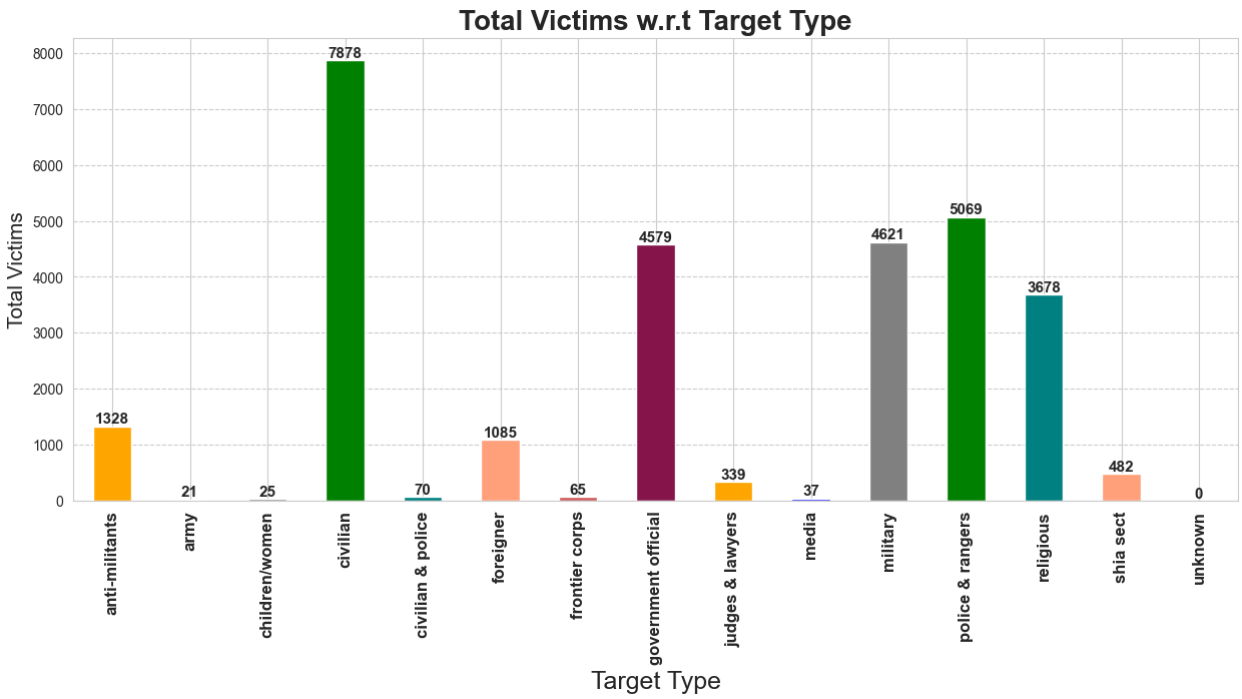
In [ ]:
```
bombari_df['Total Victims'] = bombari_df['Average Injured'] + bombari_df['Average Killed']
```

grouping:-

In [ ]:
```
grouped_data = bombari_df.groupby('Target Type')['Total Victims'].sum()
```

plotting:-

In [ ]:
```
fig, ax = plt.subplots(figsize=(15, 6))

grouped_data.plot(kind='bar', stacked=True, ax=ax, color=colors)

colors = ['orange', 'blue', 'gray', 'green', 'teal', '#FFA07A', 'indianred', '#85144b',]
sns.set_style('whitegrid')
ax.set_xlabel('Target Type', fontsize=18)
ax.set_xticklabels(ax.get_xticklabels(), fontdict={'fontsize': 12, 'fontweight': 'bold'})
ax.set_ylabel('Total Victims', fontsize=15)
ax.set_title('Total Victims w.r.t Target Type', fontsize=20, fontweight='bold')

for rect in ax.containers:
    ax.bar_label(rect, label_type='edge', fontsize=11, fontweight='bold')

ax.grid(axis='y', linestyle='--')

plt.show()
```

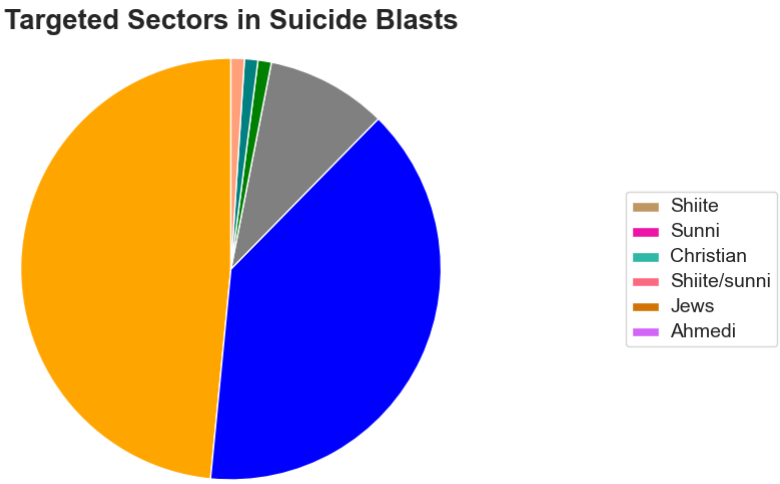


07. Suicide Blasts w.r.t Islamic Months:-

plotting:-

In [ ]:
```
import random

targeted_sects = bombari_df['Targeted Sect if any'].value_counts()
sizes = targeted_sects.values.tolist()

plt.figure(figsize=(10, 6))
plt.pie(sizes, colors=colors, startangle=90)
plt.axis('equal')
colors = ['#'+''.join(random.sample('0123456789abcdef', 6)) for i in range(len(sizes))]

table_colors = [plt.Rectangle((0, 0), 1, 1, fc=color) for color in colors]
legend = plt.legend(table_colors, targeted_sects.index.tolist(), loc='center left', bbox_to_anchor=(1, 0.5))
plt.setp(legend.get_texts(), fontsize='large')

plt.title('Targeted Sectors in Suicide Blasts', fontsize=20, fontweight = 'bold')

for label in legend.get_texts():
    label.set_fontsize(14)

print(targeted_sects)
plt.show()
```

```
Targeted Sect if any
Shiite          47
Sunni           38
Christian        9
Shiite/sunni     1
Jews             1
Ahmedi           1
Name: count, dtype: int64
```



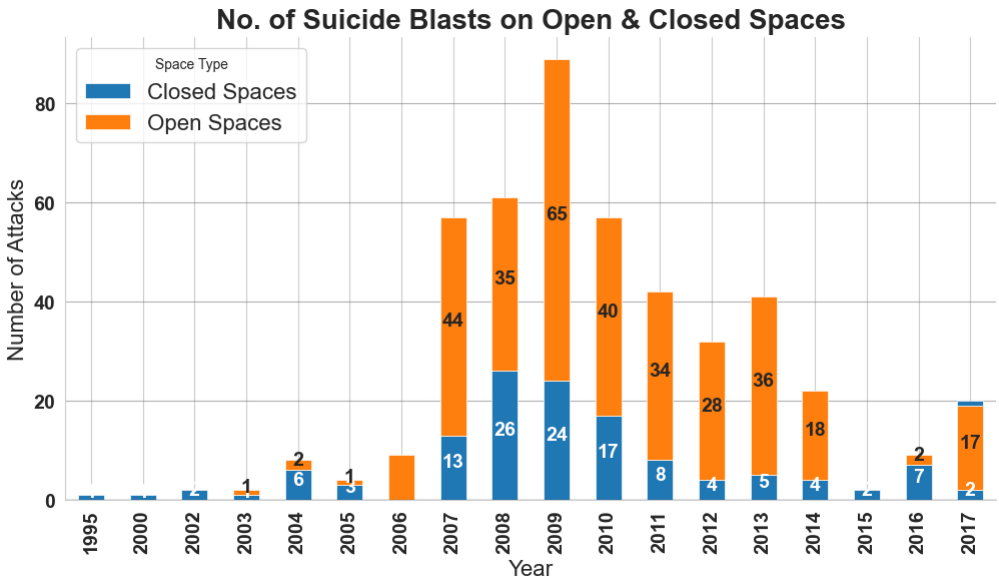
08. No. of Suicide Blasts on Open & Closed Spaces:-

plotting*:-

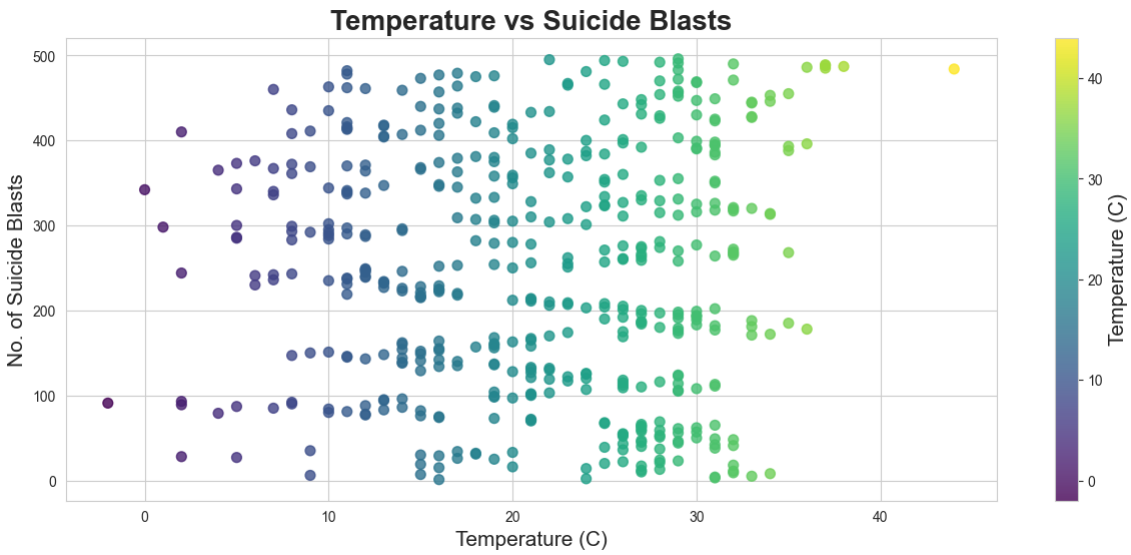
In [ ]:
```
space_counts = bombari_df.pivot_table(index='Year', columns='Open/Closed Space', values='S#', aggfunc='count')

plt.figure(figsize=(17,6))

sns.set_style("whitegrid")

ax = space_counts.plot(kind='bar', stacked=True, figsize=(12, 6), color=['#1f77b4', '#ff7f0e'], edgecolor='white', linewidth=0.5)


ax.set_title('No. of Suicide Blasts on Open & Closed Spaces', fontsize=20, fontweight='bold')
ax.set_xlabel('Year', fontsize=16)
ax.set_ylabel('Number of Attacks', fontsize=16)
plt.xticks(fontsize=14, fontweight='bold')
plt.yticks(fontsize=14, fontweight='bold')

ax.legend(labels=['Closed Spaces', 'Open Spaces'], loc='upper left', fontsize=16, title='Space Type')

ax.grid(axis='y', color='#666666', alpha=0.5)

ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

ax.axhline(0, color='black', lw=1, alpha=0.5)

for i in range(len(space_counts)):
    closed_count = space_counts.iloc[i, 0]
    open_count = space_counts.iloc[i, 1]
    ax.text(i, closed_count/2, format(closed_count, '.0f'), ha='center', fontsize=14, fontweight='bold', color='white')
    ax.text(i, closed_count + open_count/2, format(open_count, '.0f'), ha='center', fontsize=14, fontweight='bold')

plt.subplots_adjust(hspace=0.4)

plt.show()
```

```
<Figure size 1700x600 with 0 Axes>
```

```
posx and posy should be finite values
posx and posy should be finite values
posx and posy should be finite values
posx and posy should be finite values
posx and posy should be finite values
posx and posy should be finite values
posx and posy should be finite values
posx and posy should be finite values
posx and posy should be finite values
posx and posy should be finite values
posx and posy should be finite values
posx and posy should be finite values
```



09. Temperature vs Suicide Blasts:-

plotting:-

In [ ]:
```
plt.figure(figsize = (15,6))
plt.scatter(bombari_df['Temperature(C)'], bombari_df['S#'], s=50, alpha=0.8, c=bombari_df['Temperature(C)'], cmap='viridis')

plt.xlabel('Temperature (C)', fontsize=15)
plt.ylabel('No. of Suicide Blasts', fontsize=15)
plt.title('Temperature vs Suicide Blasts', fontsize=20, fontweight = 'bold')

cbar = plt.colorbar()
cbar.set_label('Temperature (C)', fontsize=15)

plt.show()
```